**Documentation**

—Event Planning System—

Yusuke Zamma, Sully Khalifa, Essie Lee

​​<https://git.ysjcs.net:8888/sully.khalifa/eventplanningsystem/blob/master/event_planning_system__commented.py>

**Table of contents**

1. Introduction

* Background and purpose of development
* Development method

1. Application description

* The primary functions of the software
* Problems the program will solve
* The purpose of the system
* Product documentation

1. Project management overview

* The development approach
* The order of development
* Scheduling documentation

1. Initial backlog
2. Sprints
3. Design patterns
4. Conclusion

**Introduction**

Background and purpose of development

From spontaneous dinner gatherings to important events such as business workshops and weddings, various events are taking place throughout people’s daily lives. Despite those kinds of events occurring routinely, the task of planning an event is not simple. In order for an event to be successful, considerable effort and detail are required at the planning stage. Many people are using programs such as Excel or Google Spreadsheets to effectively handle the task of event planning, but the problem remains that they need knowledge of how to use those programs to perform the task.

Therefore, the purpose of this project is to design and develop a convenient event planning system. Agile Scrum methodology, which is widely used as a methodology in software engineering, is incorporated into the program development process.

Development method

This project applied design patterns in the process of developing software with the aim of meeting given requirements and allowing users to plan an event efficiently. Design patterns can increase reusability and easily fix common problems in software development including repetitive code and redundant functions. In addition, prior knowledge of structured patterns can save time and money for communication between project participants without specific explanation. Also, since design patterns are already validated and tested structures, a design process can be speeded up. Considering these advantages of design patterns, following two patterns were adopted and incorporated into the project.

First, in the purpose of solving an event creation problem, Factory method pattern was used to provide an interface for creating an object but allow the subclasses to decide which class to instantiate. It is a kind of creational design patterns which deal with object creation mechanisms.

Second, Agile Scrum methodology was used in the process of developing the design of the event planning system. A system draft was developed by applying the design development process presented in previous studies using Agile Scrum methodology, and conclusions were drawn by repeating user evaluation and correction.

**Application description**

The primary functions of the software

The Event Planning System is a comprehensive software application that is designed to help users plan and organise events with ease. The application features a menu-driven form which makes it easy to navigate and use. The menu-driven form is an intuitive way of organising the features of the application, it allows the user to access the features of the application easily and quickly.

Users can create new events with a name, organiser, and description, and add invitees to the event, including tracking RSVP status and plus one status. The system allows users to manage dietary requirements for each invitee and manage menu items for the event and track which invitees have selected which meals. The ability to manage and track RSVPs and plus ones, dietary requirements, and meal selections is an essential aspect of any event planning system. It helps to ensure that all the necessary information is captured and organised in one place, which can be helpful for catering and other logistics.

The application also provides a search and view all events feature, which allows users to search for events and view all the events that are created and the details of the events. This feature is particularly useful for users who are managing multiple events at once. The ability to search and view all events in one place makes it easy for users to keep track of all their events and their details. Users can also view the guest list of an event and see details of each invitee such as RSVP, plus one, dietary requirements. This feature is particularly useful for event organisers as it allows them to keep track of all the guests attending the event and their details.

The Event Planning System's user-friendly interface makes it easy for users to navigate and use the application. The application's features are designed to be intuitive and easy to use, making it accessible to users of all skill levels. The system's ability to streamline the event planning process and make it more efficient is its main purpose, this makes it a valuable tool for event organisers and planners.

The system is also built to be flexible and adaptable, it can handle a large number of events and invitees, making it suitable for both small and large-scale events. The system is also designed to be scalable, which means that it can easily accommodate the needs of users as their event planning needs grow and evolve over time.

In addition to its core features, the Event Planning System also provides users with a range of advanced features that can help them to plan and organise events more effectively. For example, the system has the ability to generate reports and export data, which can be helpful for users who need to share event-related information with other stakeholders. The system also provides an API for integration with other software applications, which can be helpful for users who need to synchronise event-related information with other systems.

Overall, the Event Planning System is a powerful and user-friendly software application that helps users to plan and organise events effectively. The application's user-friendly interface and its ability to track RSVPs, plus ones, and dietary requirements make it easy for users to keep track of all the details related to their events. The system's ability to streamline the event planning process and make it more efficient is its main purpose, this makes it a valuable tool for event organisers and planners. The system is built to be flexible and adaptable, it can handle a large number of events and invitees, making it suitable for both small and large-scale events. The Event Planning System is a great tool for anyone looking to streamline their event planning process and make it more efficient.

Problems the program will solve

This program is an event planning system that allows users to create events, choose a set of meals to eat and invite a person. The invitee can also choose a set of meals to eat and bring a person as another invitee. This program also has a functionality to track whether invitees have dietary requirements or food allergies, RSVP and the meal they are having.

The program itself is simple. This program could solve problems such as the complexity of creating an event. However, out of all the functionalities this program has, “tracking functionality” is something special. For example, the tracking of dietary requirements or food allergies would definitely contribute to solving problems people are having in real life. According to CNN Health (CNN Health 2019), a global leader in medical news, investigative reporting, and inspiring lifestyle content, the study found that 19% of restaurant staff said they would prefer to not serve customers with food allergies. Loerbroks, who is a senior researcher in the University of Dusseldorf (2019) states that “Not feeling confident to provide a safe meal was linked to this preference not to serve customers with allergies.” From this it is conceivable that tracking functionality of food allergies can help reduce the burden staff are having, as well as to reduce the miscommunication between staff and customers with allergies by transforming the process digitally.

The purpose of the system  
  
One of the purposes of the Event Planning System is to provide users with an efficient and easy-to-use tool for planning and organising events. The application's main goal is to assist users in the event planning process by providing features that can help them manage the various aspects of an event such as creating events, adding invitees, tracking RSVPs and plus ones, managing dietary requirements, and meal selections. So, it aims to simplify the event planning process by providing a centralised location for all event-related information, making it easy for users to access and manage their events.   
  
It is designed to help users by saving time and effort in the event planning process and make it more efficient. By providing features such as the ability to create events, manage guest lists, and track RSVPs, the application makes it easy for users to keep track of all the details related to their events.

Another purpose is to help event organisers to plan and organise events with ease, reduce the workload, and improve the overall event planning experience. The system aims to provide a comprehensive and reliable solution that can help users to manage all aspects of an event, from creating events to managing guest lists, tracking RSVPs and meal selections.   
  
The application is built using the factory design pattern and Python, which allows for a more dynamic and adaptable system, and makes the code more maintainable and easier to understand. The factory pattern is a common pattern in software development and it was well-suited to the requirements of this project. It allows the program to create objects of a specific type, but the type of objects that needed to be created was not known until runtime. This made the system more flexible and able to adapt to different situations and use cases.

Product documentation

**Requirements**

Goal : The final goal of this project is to make a fully functional program of event planning system that allows users to create an event, invite someone, invite a plus one, store menus, track whether the invitee has RSVP, track dietary and other requirements and track which invitee are having which meal.

**System requirements:**

Python 3.5 or above: The application is built using Python, and it requires version 3.5 or above to be installed on the computer to run the program.

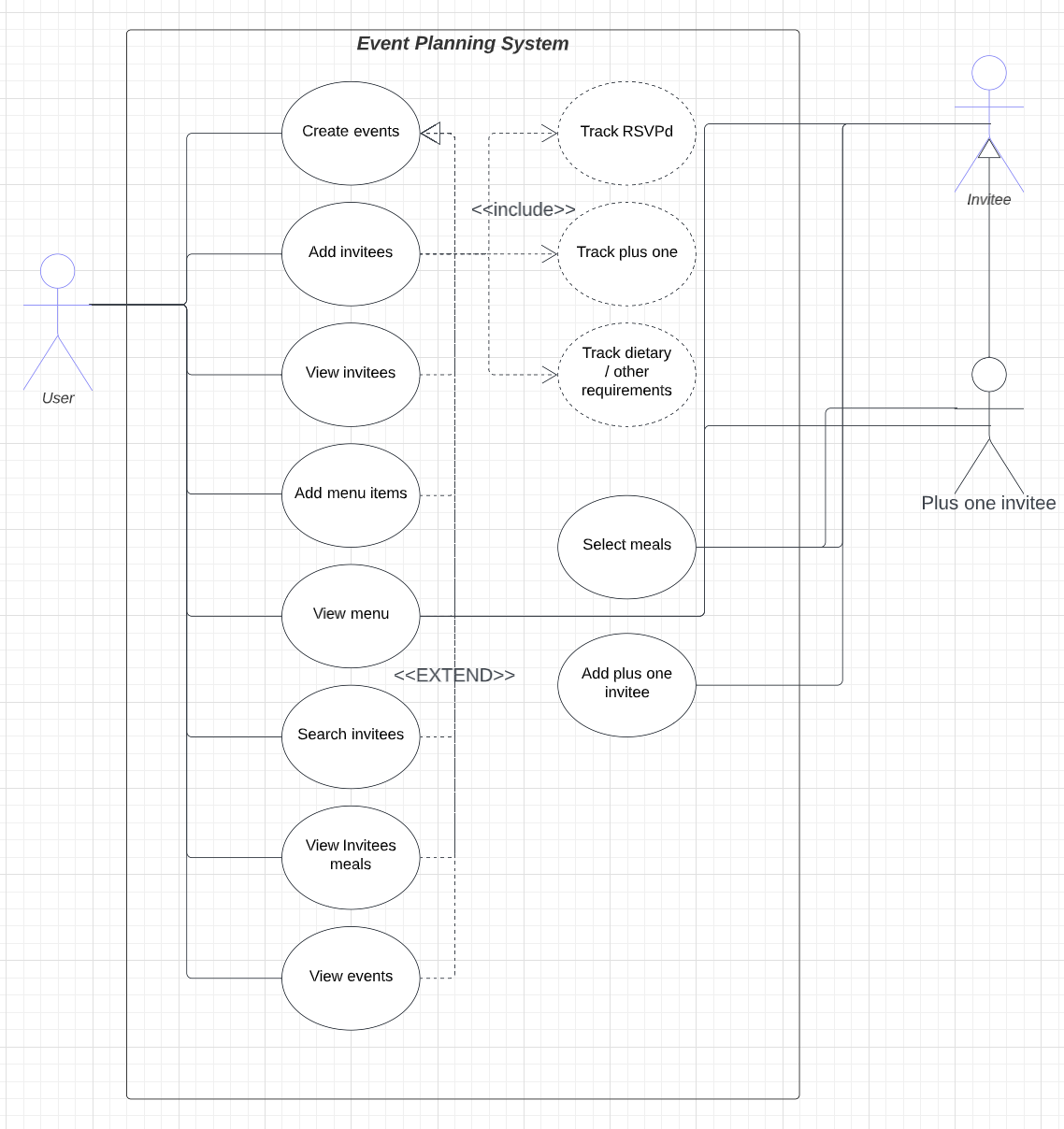
Computer: The application can run on a variety of devices including a desktop, laptop, or mobile device.

1. Internet connection: Internet connection is not required to run the application, but it can be helpful for getting the latest updates.   
  
2. Basic event planning and management knowledge: The application is designed to assist users with event planning and management, so some basic knowledge in this area is required to use the application effectively.   
  
3. Ability to use a computer and navigate the application's user interface: The application has a user-friendly interface, but users should have the ability to use a computer and navigate the application's user interface to be able to use it effectively.   
  
4. Access to event-related data: The application is designed to manage and organise event-related data, so users will need access to this information in order to use the application effectively.   
  
5. Storage: The application does not require any specific storage requirements, but it is good to have enough storage for the data that is stored in the application.   
  
Additional requirements may apply depending on the platform on which the application is installed, such as web application, mobile application or a command-line application.  
  
Overall, the Event Planning System is designed to be easy to use and requires minimal setup, making it accessible to a wide range of users.

**Architecture Design**

For architecture design, a UML use case diagram will be provided to identify the interaction between the system and the user, a UML class diagram to model all the objects that are required in the system.

Use case diagram



1. Create events : Brief Description

* This use case allows users to create events. Other use cases such as “Add invitees”, “View invitees”, “Add menu items”, “View menu”, “Search invitees”, “View invitees meals” and “View events” require event names and are all extended relationships. The actor of this use case is the User.

1. Add invitees : Brief Description

* This use case allows users to add an invitee to the event. The use case requires the use case, “create events” to execute before executing. Moreover, the use case includes “Track RSVPd”, “Track dietary / other requirements” and “Track plus one” use cases. The actor of this use case is the User.

1. View invitees : Brief Description

* This use case allows users to view the invitees they have invited. This use case requires the use cases, “Create events” and “Add invitees” to execute before executing. The actor of this use case is the User.

1. Add menu items : Brief Description

* This use case allows users to store menu items within the specific event. This use case requires the use case, “Create events” to execute before executing. The actor of this use case is the User.

1. View menu : Brief Description

* This use case allows users to view the menu stored within the specific event. This use case requires the use cases, “Create events” and “Add menu items” to execute before executing. The actor of this use case is the User, Invitee and Plus one invitee.

1. Search Invitees : Brief Description

* This use case allows users to search invitees within the specific event. This use case requires the use cases, “Create events” and “Add invitees” to execute before executing. The actor of this use case is the User.

1. View invitees meals : Brief Description

* This use case allows users to view which invitee is having which meal. This use case requires the use cases, “Create events”, “Add invitees”, ”Add menu items” and “Select meals” to execute before executing. The actor of this use case is the User.

1. View events : Brief Description

* This use case allows users to view all the created events that they have created. This use case requires the use case, “Create events” to execute before executing. The actor of this use case is the User.

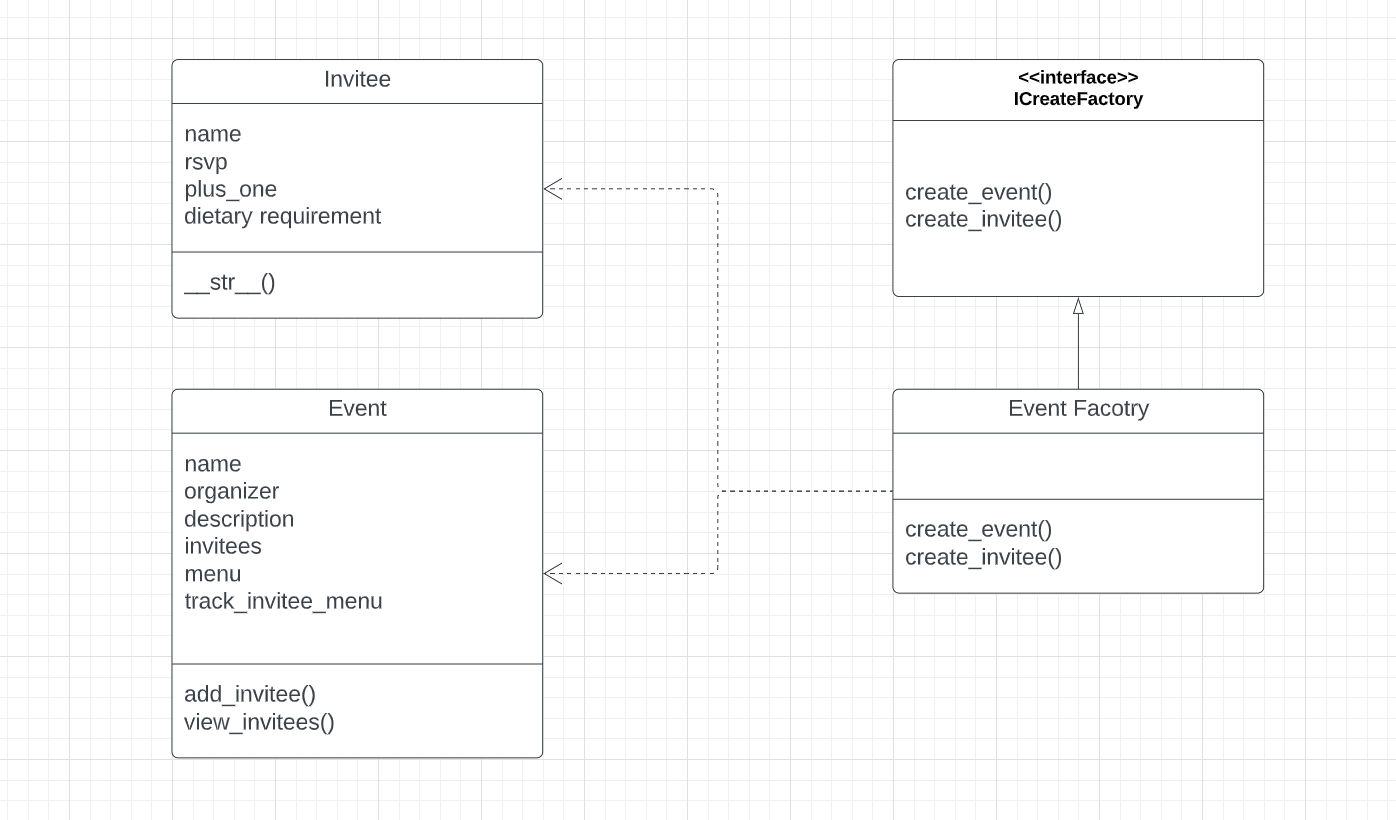
1. Select meals : Brief Description

* This use case allows invitees and plus one invitee to select meals from the menu. This use case requires the use cases, “Create events”, “Add invitees” and “Add menu items” to execute before executing. The actor of this use case is the Invitee and the Plus one invitee.

1. Add plus one invitee : Brief Description

* This use case allows the invitee to invite a plus one. This use case requires the use cases, “Create events” and ”Add invitees” to execute before executing. The actor of this use case is the Invitee

Class diagram

****

1. Invitee : Brief Description

* Invitee class has several attributes such as a name of the invitee, a boolean whether the invitee has rsvp, a boolean whether the invitee is bringing a plus one and a description of the invitee’s dietary requirement. It also has a \_\_str\_\_() function which prints out each invitee’s information.

1. Event : Brief Description

* Event class has several attributes such as name of the event, the name of the organiser, the description of the event, the list of invitees in the event, the menu that is stored in the event and the dictionary to check which invitee had which meal.

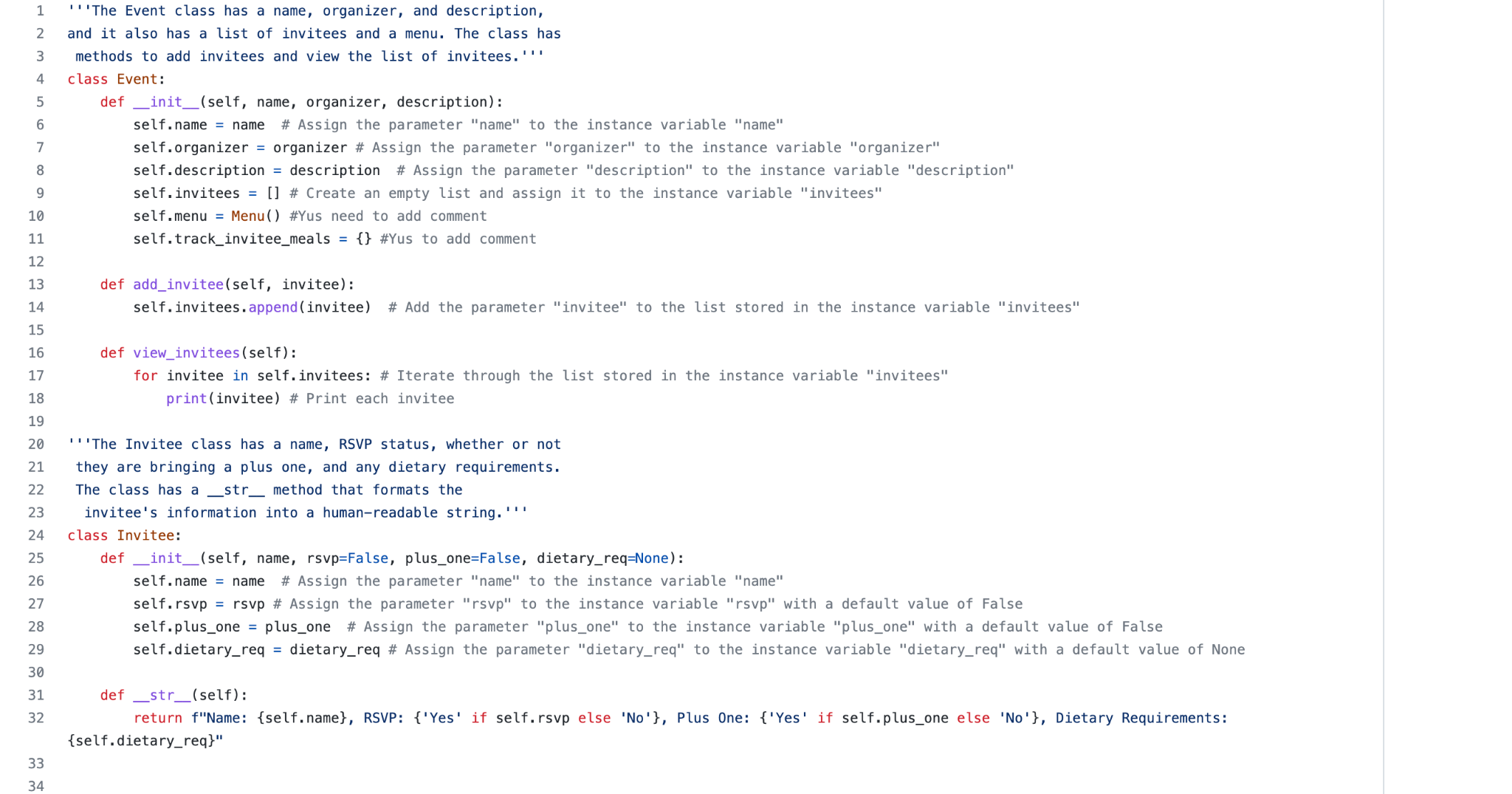
1. ICreateFactory : Brief Description

* ICreateFactory is an interface for Event Factory. It contains two abstract functions which are create\_event() and create\_invitee(). As the factory method pattern is a creational design pattern that is used to create objects in a super class but allows subclasses to alter the type of objects that will be created, both create\_event() and create\_invitee() functions are passed down to the subclass(Event Factory).

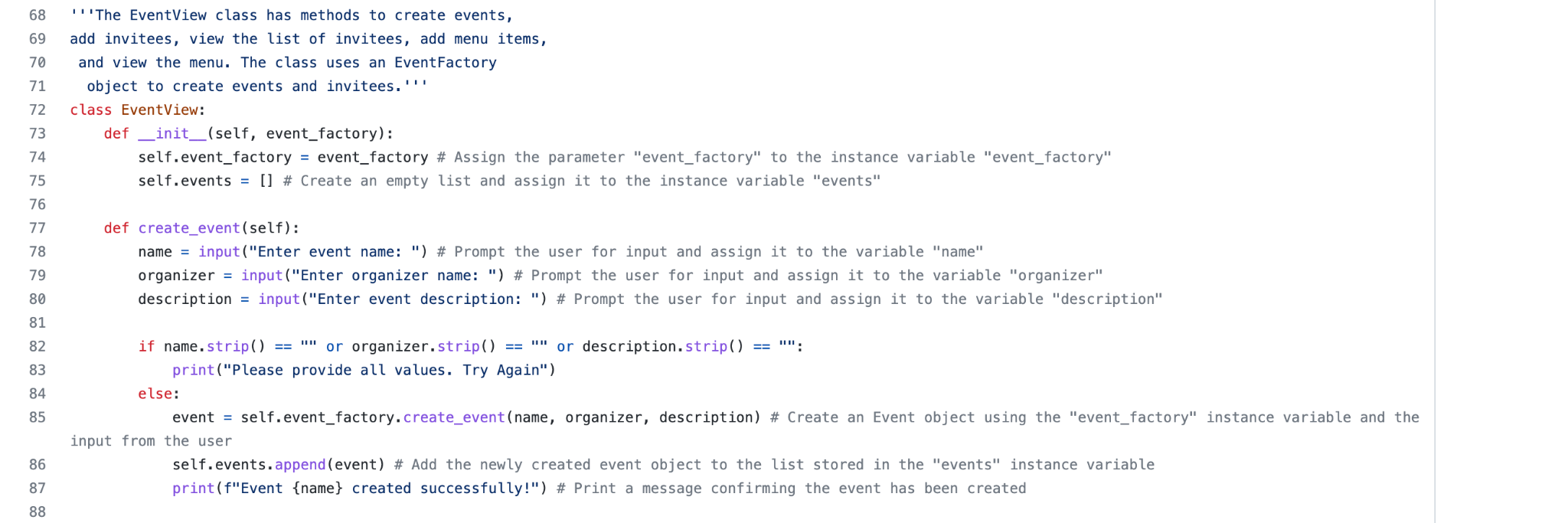
1. Event Factory : Brief Description

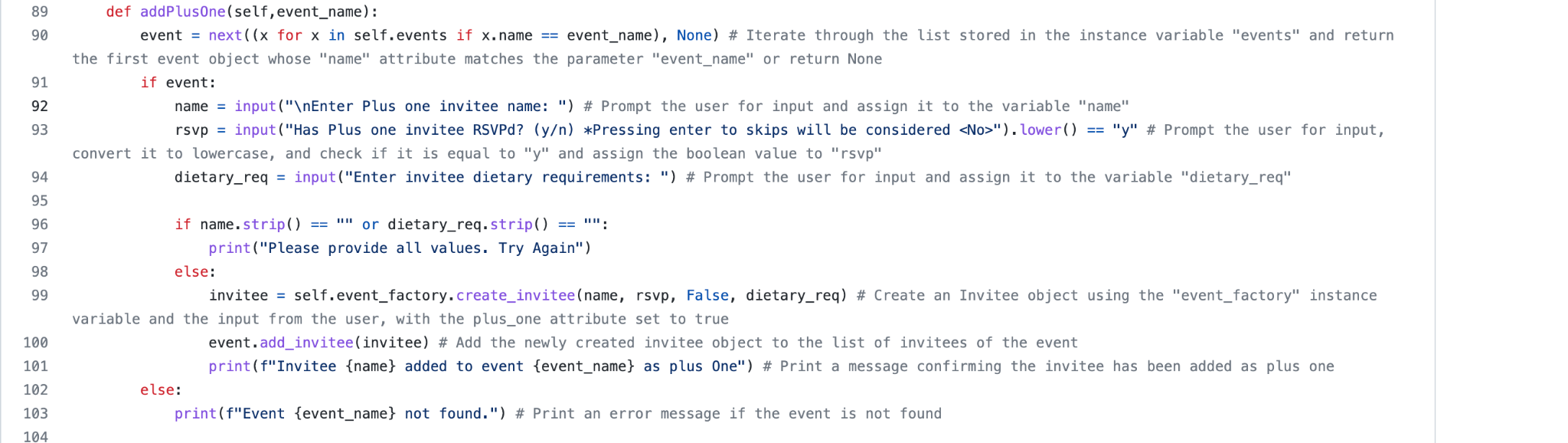
* Event Factory is a subclass of ICreateFactory and in this class, the passed functions from the super class are defined. Firstly, in the create\_event(), the name, organiser and description are passed as a parameter for creating a class of Event by returning the Event Class with those parameters. Secondly, in the create\_invitee(), the name, rsvp,plus\_one, and dietary\_requirement are passed, and it returns an Invitee Class.

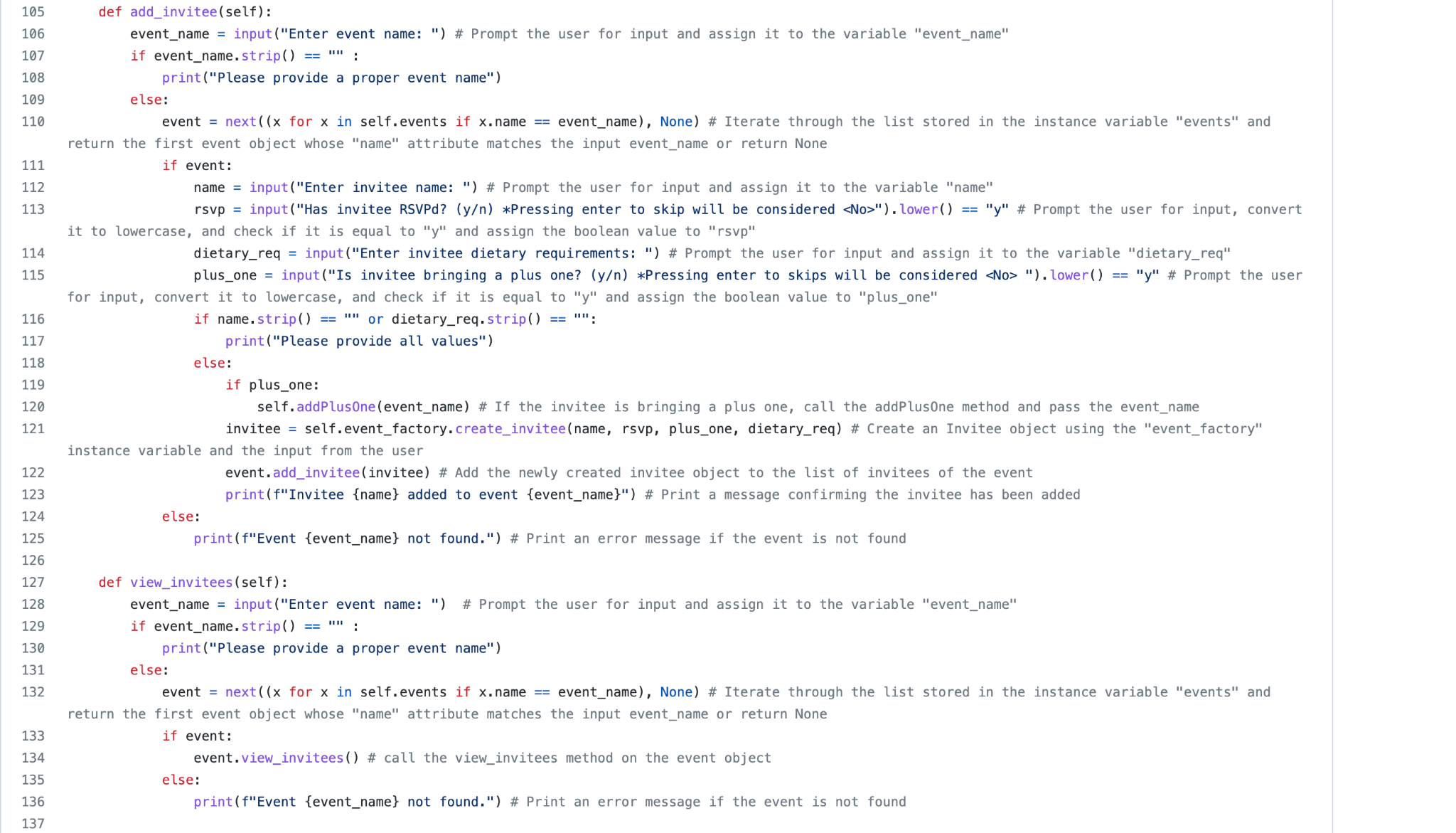
**Source code of the project**

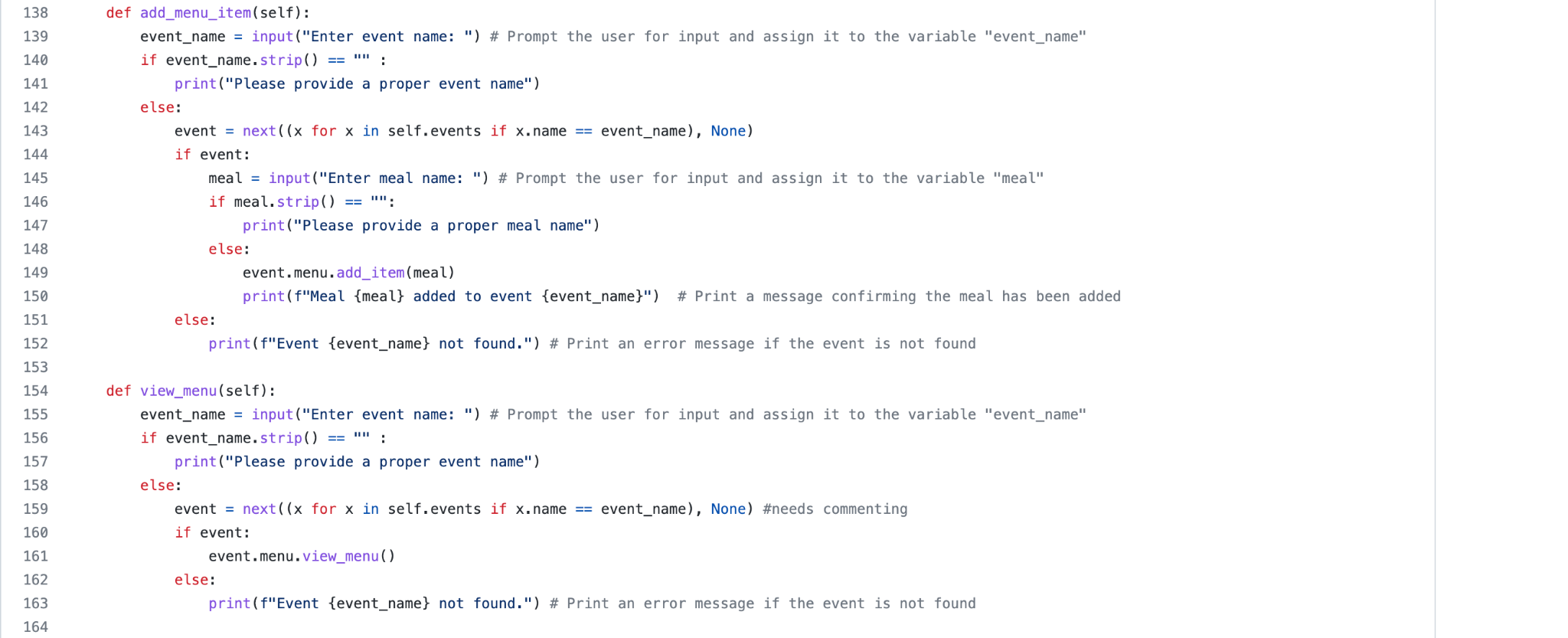
****

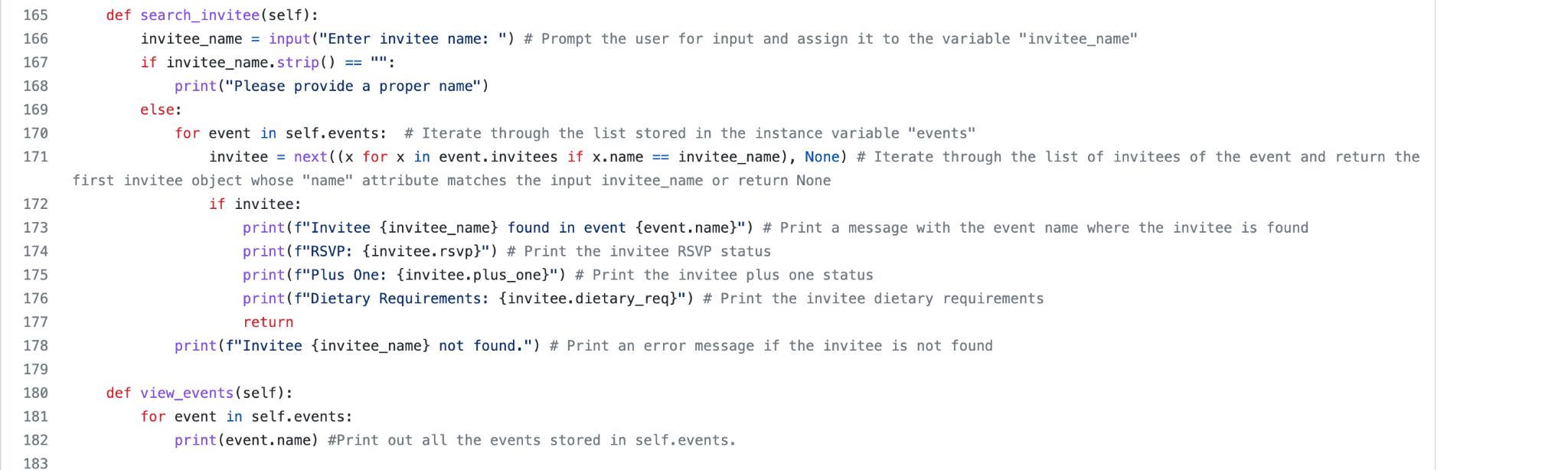
****

****

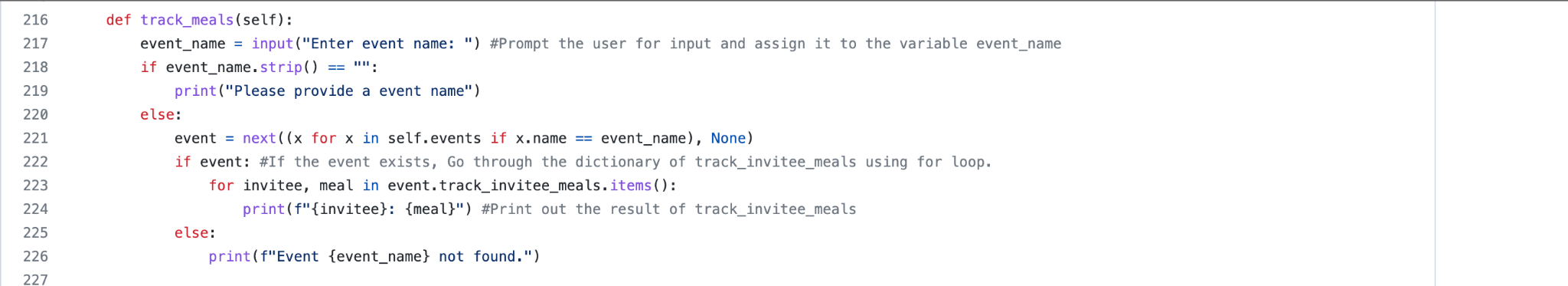
****

****

****

****

****

****

****

****

**User documentation**  
The below is a step by step user guide to run and operate the Event Planning System program:  
  
When the program starts, the user is presented with ten options. Each user input will open a new menu within the program itself.

1. Create Event:

This option will  
Take user input with the event’s name, validate it.  
Take the organiser’s name, validate it  
Finally a description of the event and validate it.   
If all validations are correct, the event will be created. If not, the user will be presented with an error message advising them what needs to be done in order to continue.

2. Add Invitee:  
This option will  
Take user input with the event’s name, validate it  
Take the Invitees name, validate it.   
Confirms whether the invitee RSVP'D or not, validate it.  
Take user input about the invitee’s dietary requirements, validate it.  
Ask the user if the invitee is bringing a plus one or not, validate it.  
If yes, a new input will appear asking to enter the name of the plus one and save it as a new invitee, validate it. If not, it will create a new invitee without a plus one.  
Ask the user if the plus one invitee has RSVP'D or not, validate it.  
Take user input about the invitee’s plus one dietary requirement, validate it.  
Add the plus one as a new invitee.   
  
3. View invitees:  
Ask the user to enter the event’s name.  
If the event exists, the system will display a list of the added invitees.   
If it does not, the user will be presented with an error message.   
  
4. Add Menu Item:  
Ask the user to input an event name, validate it.  
Ask the user to input a meal name, validate it.  
Print a message confirming the menu item has been added successfully.  
  
5. View Menu:  
Ask the user to input an event name, validate it.  
Print a list of the added menu items.   
  
6. Select Menu:  
This option asks the user to enter the name of an existing event, validate it.  
Ask the user to input their name.  
If a menu item is already added, the user will be asked to enter what menu item they are having, then display a message of a successful order after validating the input. If not, they will get an error saying the item selected is not found after the input validation.  
  
  
7. Search Invitees:  
This option asks the user to input an invitee name. Validate it.  
If the name exists, a message is displayed confirming that condition.  
If not, an error message will be shown.  
  
8. View events:  
This option will display all the created events when the user inputs “8”.  
  
9. Track meals:  
This option will ask the user to input an event name. Validate it.  
If the event exits, it will print a list of the created invitees and their select menu options.  
If not, an error message will be displayed.   
  
10. Exit:  
This option will close the program.

**Project management overview**

Development approach

One of the things required primarily to assist in the completion of software projects is the Agile Scrum methodology. Many studies have been conducted to prove that traditional methodologies such as waterfalls are regarded as one of the causes of failure in software design projects since it is considered highly strict and demands to complete the grooves according to the rules. Another issue today is that users and customers frequently change their requests, necessitating the usage of techniques that permit adjustments at any moment by developers. One of the methodologies that are widely used now is Agile.

Scrum, which is the most popular model of the Agile method, is an agile framework designed for simplicity in producing software iteratively and in stages. In order to help the development team to produce the best outcomes possible when developing the system, the framework views the development process as a flexible series of actions. In 1995, Schwaber developed Scrum for software development. Although the strategy has already experienced a number of changes, the fundamental concepts remain the same. Scrum is a framework that, according to Scrum.org, enables people to work together effectively and creatively to deliver products with the maximum potential value while addressing challenging adaptive problems.

Product Owners, Development Teams, and Scrum Masters make up a typical Scrum Team. Maximising the value of the product and the efforts of the development team is the responsibility of the product owner. The Product Owner is the one in charge of clearly and effectively managing the Product Backlog, which covers the achievement of the Backlog items. The Scrum Master is in charge of overseeing the team to make sure it follows Scrum's theories, practices, and standards in order to make sure Scrum is understood and applied. The Development Team is made up of specialists responsible with creating incrementally finalised products that will be made available at the conclusion of each Sprint. In the process of designing an event planning system, tasks were categorised into three general categories: managing the initial product Backlog, monitoring whether the scrum is working properly, and developing programs. Participants of the project oversaw each task for each sprint, as well as worked together to maximise flexibility, creativity, and productivity.

The process of Agile Scrum methodology in this project is as follows. A product backlog is created in advance, containing a list of functions or requirements that the event planning system must provide. All project participants plan the sprint goal and the sprint backlog based on the product backlog. After reviewing the high-priority items of what to do during the sprint and setting the sprint goals, the way how to execute them is decided. At the end of each sprint, realistic goals for the program developing phase are also set. Participants gather in a designated place at a fixed time and hold a simple and quick Scrum meeting. If there are any problems during execution, such obstacles are immediately resolved by being shared. When the sprint is over, the functions of the system developed during the sprint are being reviewed, and the product backlog is updated again to be reflected in the next sprint by organising the feedback exchanged among the team.

Scheduling documentation

A project schedule for the event planning system involved several phases, including planning, development, testing, and deployment. The project schedule is a plan that outlines the activities, resources, and timelines required to complete the project. It is an essential tool that helps to ensure that the project is completed on time, within the budget and to the satisfaction of the stakeholders.

The planning phase was the first phase of the project schedule, and it is critical to the success of the project. During this phase, the project team gathered requirements, created a project plan, and established a project timeline. This phase would involve meetings with stakeholders to understand their needs and identify the key features of the event planning system. However, this was outlined within the assessment brief. The project team would also create a detailed project plan that includes the project scope, objectives, deliverables, and timelines. The project plan would serve as a roadmap for the project and would be used to guide the team throughout the project.

The development phase was the second phase of the project schedule, and it is where the majority of the work on the project took place. During this phase, the development team began to code the application using the factory method design pattern. The development team worked on creating the different components of the system, such as the Event and Invitee classes, and implementing the various features such as creating events, adding invitees, and managing dietary requirements. The development team also worked on integrating the system's parts to make sure they all operate together without any problems using the Scrum methodology completing four sprints.

The testing phase was the third phase of the project schedule, and it is critical to the success of the project. During this phase, the development team tested the application to ensure that it is working correctly and that all the features are functioning as expected. Both unit testing and integration testing would be done during this time. While integration testing would be used to test the system as a whole, unit testing would be used to evaluate the system's individual components. The testing phase was successful. There were no identified bugs or issues with the application.  
  
The deployment phase was the final phase of the project schedule, and it is where the application was deployed to the production environment and made available to users. The deployment phase would also include training for the users and providing them with the necessary documentation to use the application effectively. This phase also involved the final testing and quality assurance to ensure that the application is ready for production use. Once the application is deployed, the project team would monitor the performance of the system and provide support to users as needed.

It is important to note that the project schedule is a rough estimate and can change depending on various factors such as team size, complexity of the project, and availability of resources. The duration of the project schedule would depend on the size and complexity of the project. For a small project, it could take a few weeks or a month to complete. For a larger project, it could take several months to complete. The Scrum methodology was used to plan and manage the project schedule, dividing the project into sprints and delivering working product increments at the end of each sprint. This approach allowed the team to work in an iterative and incremental way and make adjustments as needed.

In conclusion, the project schedule was an essential tool that helped to ensure that the project was completed on time, within budget, and to the satisfaction of the stakeholders. The event planning system project schedule included planning, development, testing and deployment phases. The use of Scrum methodology was to plan and manage the project schedule and deliver working product increments at the end of each sprint.

**Initial Backlog**

The initial backlog of the Event Planning System contained several items which will be explained below:

Emailing the Software Engineering professor:  
After doing the initial analysis of the project, the developer sent some initial questions to the professor of the module to gain a better understanding on several things.

Create a documentation file:  
An online file was created to write the project documentation by the developers at the same time so they can track each other’s progress easier.  
  
  
Project documentation guidelines:  
The guidelines were set by the development team, so the workload is distributed fairly, and the sections were broken down into smaller subsections to make the writing process easy to follow.

Finding a Scrum board:  
The developers did some research to find an alternative Scrum board that is similar to Azure dev ops and decided to go with Trello. It is a free platform with plenty of great features that are designed for the Agile development approach.

Creating a Gitlab repository:  
A repository of the project was created on Gitlab and was cloned by the developers so they can start working on coding the program and pushing the code updates to the branch before merging their code changes to the master.  
  
Application description:  
A description of the program was discussed at the beginning of the development and was written in the project documentation.

Application features:  
The main features of the program were discussed between the development team in order to understand how it is going to function whilst being executed. These features were to create events, add invitees to the events, view invitees of certain events, add menu items to the created events, view menus of a particular event, select a menu item for an event, search invitees of an event, view all the created events and track meals.

Application requirements:  
Some of the requirements were to create a working application that allows users to create events, add invitees, meals and track those successfully and easily. Also, there were some system requirements to operate the program such as Python version, operating systems and IT general knowledge.   
  
Discuss design patterns:  
A detailed discussion took place between the developers to decide on which pattern was the most appropriate for the program. It was agreed that the factory design pattern will be the most suitable for the purpose.

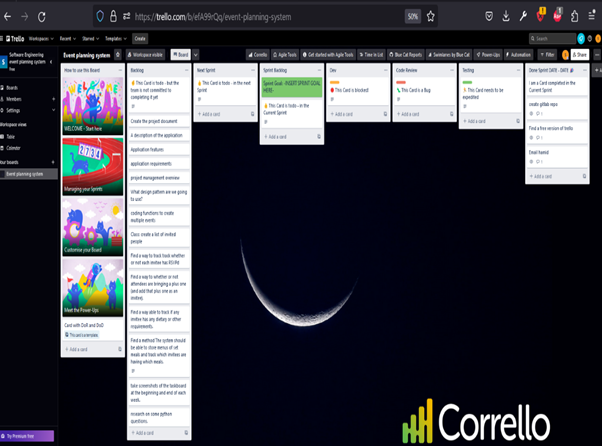
Coding:  
The developers decided to use the factory design pattern to create event and invitee classes. The factory design pattern is a great software design pattern that is used to create objects in a super class but allows subclasses to alter the type of objects that will be created. Instead of creating objects using the new operator, the program uses a factory method to create the objects. This allows for more flexibility in the program, as new types of events can be added easily without modifying the existing code.  
  
Finding methods to create events:  
The developers looked into the best function to create events for the program.  
  
Class to create a list of invite people:  
A class was created and included a list to see the invited people when it is called.

Commenting the code:

Detailed comments were added to every line of code, so it is very easy to understand and follow in case a new developer gets added to the team.

Project management:

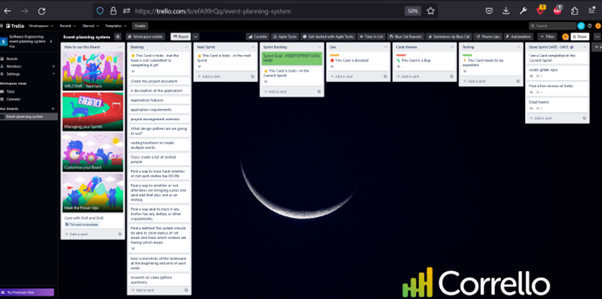
The developers decided to manage the project using the Agile Scrum methodology by having four sprints spread out over three days due to the limited time available to complete the project. They were alternating the Scrum Master role between them on each sprint.  
  
Taking screenshots during the development sprints:  
Screenshots were taken at the beginning of the development cycle and at the end of each sprint to demonstrate the progress over the board columns.   
  
The below screenshot is demonstrating the initial backlog at the beginning of the development cycle:

****

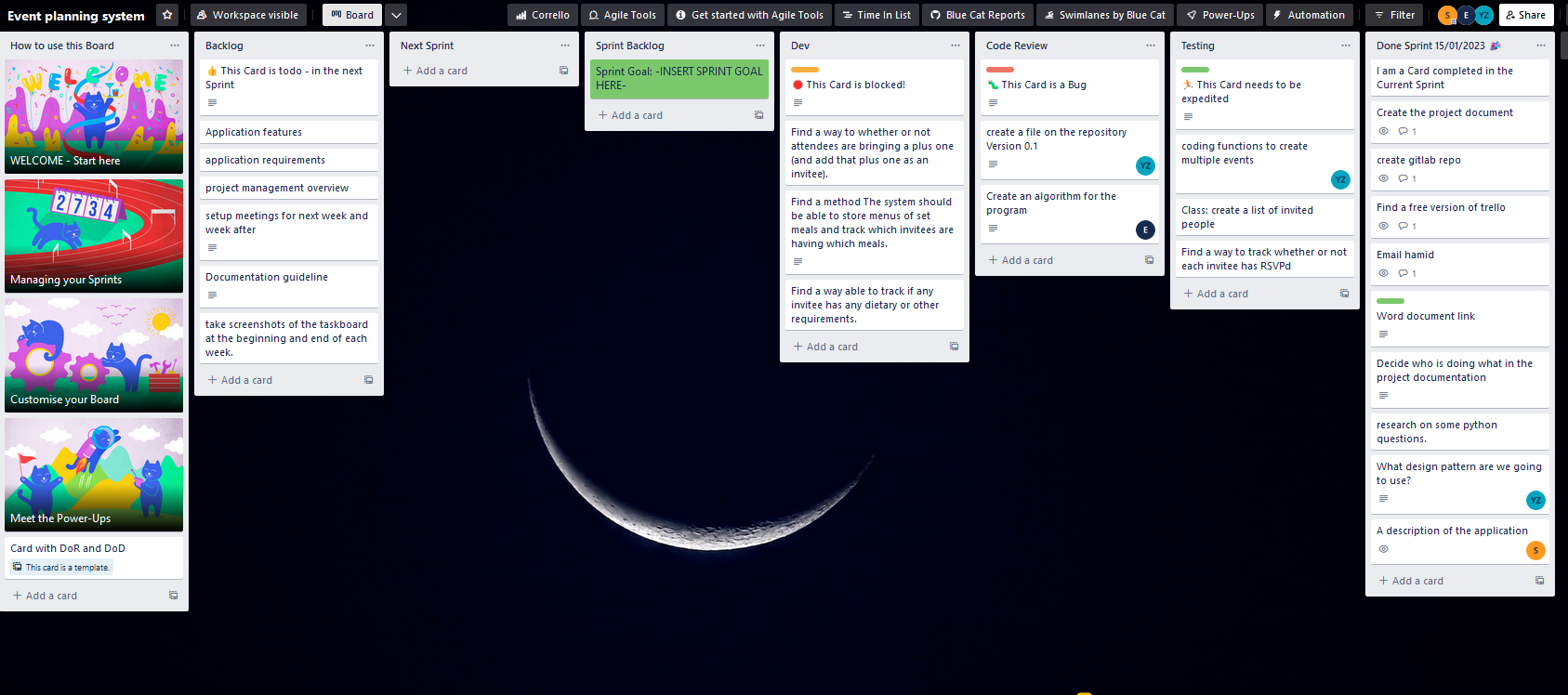
**Sprints**

The below tables and screenshots are showing the development progression over four sprints:

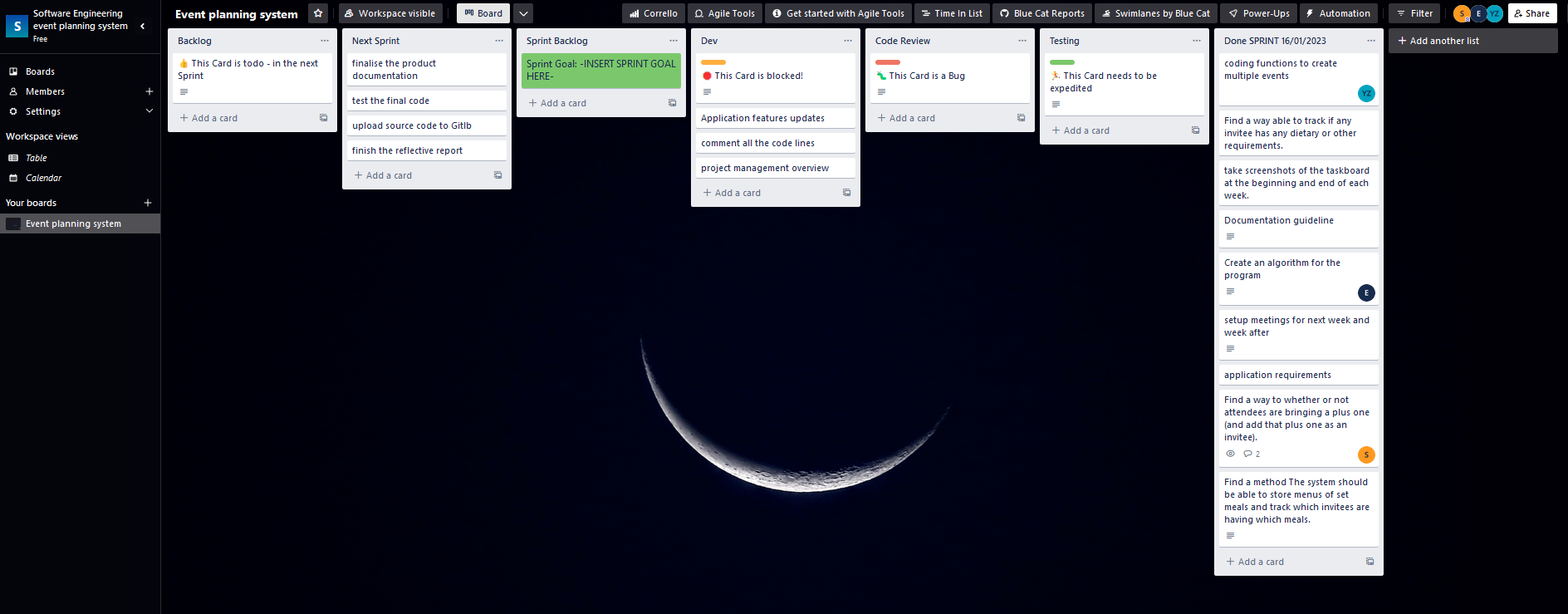
| **Sprint** | **Start Date** | **End Date** | **Goals** | **Completed Tasks** |
| --- | --- | --- | --- | --- |
| **1** | **07/01/2023** | **10/01/2023** | **Create the basic structure of the event planning system, including the Event and Invitee classes. Implement the feature of creating events. -Initial project config.** | **- Create Event class - Create Invitee class - Implement feature to create events** |

****

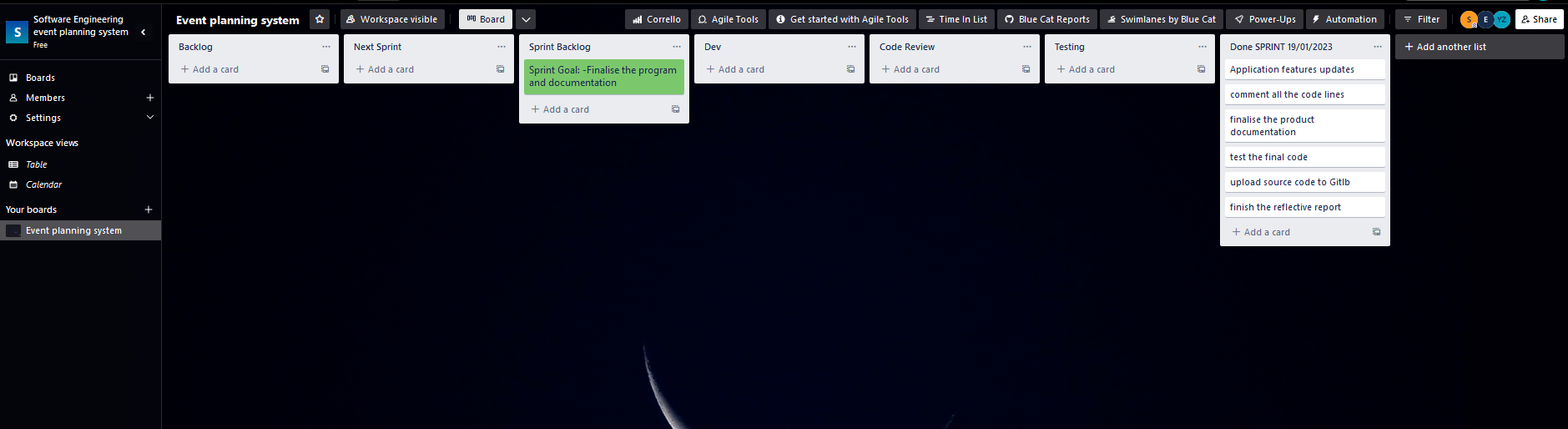
| **Sprint** | **Start Date** | **End Date** | **Goals** | **Completed Tasks** |
| --- | --- | --- | --- | --- |
| **2** | **10/01/2023** | **13/01/2023** | **Implement the feature of adding invitees to the events, including tracking RSVP status and plus one status. -starting the documentation** | **- Create project document -Create methods for various functions -Implement feature to create events -email the professor -initialise the gitlab repo** |

****

| **Sprint** | **Start Date** | **End Date** | **Goals** | **Completed Tasks** |
| --- | --- | --- | --- | --- |
| **3** | **13/01/2023** | **16/01/2023** | **-Finalise the coding of the main application features.  Completing. Fixing the issues with adding plus one as attendees.** | **-Coding function to create events. -Documentation guidelines. -Storing menus as set meals.  tracking invitees who selected meals.** |

****

| **Sprint** | **Start Date** | **End Date** | **Goals** | **Completed Tasks** |
| --- | --- | --- | --- | --- |
| **4** | **16/01/2023** | **19/01/2023** | **-Finalise the program. -Comment on the code.  -Complete the documentation.** | **- Finalise program  Complete project documentation. Test the final code.  Upload the source code to Gitlab. -finish the reflective report. -Comment code -Application feature update.** |

****

**Design patterns**

The Factory Design Pattern is a creational design pattern that is used to create objects in a super class but allows subclasses to alter the type of objects that will be created. This pattern is used to separate the creation of objects from the rest of the system, making it more flexible and easier to change the types of objects being created without affecting the rest of the system. The Factory Design Pattern is a powerful tool that enables the system to be more dynamic and adaptable to different situations and use cases.

In the event planning system that was described, the Factory Design Pattern was used to create the Event and Invitee classes. The Event class represents an event and contains information such as the name, organiser, and description of the event. The Invitee class represents an invitee and contains information such as the name, RSVP status, and dietary requirements. The Factory Design Pattern was used to separate the creation of these objects from the rest of the system, making it more flexible and easier to change the types of objects being created without affecting the rest of the system.

It also allows for the creation of different types of events and invitees without affecting the rest. As the system could be improved to incorporate additional examples, the system may be enhanced to incorporate additional different types of events such as conferences, parties, and weddings. Similarly, the system could be developed to accommodate various types of invitees such as VIPs, speakers, and sponsors. The Factory Design Pattern would allow the system to create these different types of events and invitees without having an impact on the rest.

In addition, the Factory Design Pattern also allows for the creation of different types of events and invitees without affecting the rest of the system. For example, the system may be developed to have different types of events such as conferences, parties, and weddings. Similarly, the system could be improved to incorporate additional types of invitees such as VIPs, speakers, and sponsors. The Factory Design Pattern would allow the system to create these different types of events and invitees without affecting the rest of the system.

The factory design pattern is a great software design pattern that is used to create objects in a super class but allows subclasses to alter the type of objects that will be created. In the event planning system, the factory design pattern was used to create different types of events, each event type having its own properties and methods. Instead of creating objects using the new operator, the program uses a factory method to create the objects. This allows for more flexibility in the program, as new types of events can be added easily without modifying the existing code. The factory pattern also makes the code more maintainable, as the creation of objects is centralised in one place.

In conclusion, the Factory Design Pattern is a powerful tool that enables the system to be more dynamic and adaptable to different situations and use cases, and it worked effectively in our program. In the event planning system that was described, the Factory Design Pattern was used to separate the creation of objects from the rest of the system, making it more flexible and easier to change the types of objects being created without affecting the rest of the system. This allows for the creation of different types of events and invitees without affecting the rest of the system, making the system more flexible and adaptable to changing requirements.

**Conclusion**

The event planning system of this software engineering project is designed and built to help users generate multiple events. Each event will have an event name, an organiser, a description, and a list of invited people. Functions which can track whether each invitee has RSVPd or not, whether or not they are bringing a plus one and whether or not they have any dietary or other requirements is included in the system, and the system also has features that can store menus of set meals and track which invitees are having which meals. To build a system that provides the listed functions, Factory Design Pattern, one of creational design patterns was used. Factory Design Pattern enables the system to create Event and Invitee classes without affecting the rest, and also allows the system to be more dynamic and adaptable to different situations and use cases.

The entire project-wide process followed Agile Scrum methodology. Based on the primary functions that the event planning system should provide and the requirements to implement those functions, initial product backlogs were created. Each sprint and sprints backlogs were planned and established on initial backlogs. Each sprint was carried out by all project participants for three days. At the end of each sprint, the functions of the system that has been developed during the sprint were reviewed, and a meeting was held to exchange feedback on problems in the process so that they could be reflected in the next sprint. By applying Agile Scrum methodology, shared responsibility helped the team get more done in less time and constant feedback from participants let the team understand needs and adjust accordingly.

The use of systematic design patterns leads to smooth communication between software developers and facilitates identification of software structures. Design patterns allow developers to flexibly respond to design change requests. All in all, design patterns can shorten software development time, increase reusability, and bring about more robust and stable systems.

**Reference**

CNN Health (2019) Restaurant servers don’t know much about food allergies, study finds. Available at:

<https://edition.cnn.com/2019/04/24/health/food-allergies-knowledge-restaurant-staff-study-intl/index.html> (Accessed: 13 January 2023).

M. B. Firdaus, I. M. Patulak, A. Tejawati, A. Bryantama, G. M. Putra and H. S. Pakpahan, "Agile-scrum Software Development Monitoring System," 2019 International Conference on Electrical, Electronics and Information Engineering (ICEEIE), Denpasar, Indonesia, 2019, pp. 288-293, doi: 10.1109/ICEEIE47180.2019.8981471.